AIAA-2009-3990
19th AIAA Computational Fluid Dynamics Conference, San Antonio, Texas, June 22-25, 2009.

Enhancements to the Hybrid Mesh Approach to Surface Loads Integration on Overset Structured Grids

William M. Chan*
NASA Ames Research Center, M/S T-27B, Moffett Field, CA 94035

Accurate computation of forces and moments on structured overset surface grids requires careful accounting of the overlapped zones. The advantages and disadvantages of two methods in current practice, the hybrid mesh approach and the weighted panels approach, are discussed. New schemes that enhance the software execution speed and robustness of the hybrid mesh approach are presented. Use of index-space-balanced bounding boxes for stencil searches, and various other optimization techniques resulted in a speed up in code execution time by a factor of about 40. Robustness of the triangulation scheme is also improved by a boundary sub-string splitting procedure and a new sequence of triangular cell construction tests. Triangle surface normal consistencies are established using logical tests based on index space directions rather than using floating-point arithmetic checks.

I. Introduction

STRUCTURED overset grids have been used in high-fidelity simulations for a wide variety of aerospace vehicles since the early 1990's.[1–8] For many of these computations, the ultimate critical goal is accurate prediction of the aerodynamic forces and moments for the entire vehicle as well as for its various components. Such information is extracted from the numerical solutions by integrating pressure and viscous stresses over the surface grids. Since the surface grids in a structured overset configuration are allowed to overlap arbitrarily, careful accounting of the overlapped zones is crucial in accurate loads computation.

The first attempt to address the issue of accurate overlapped zone accounting was based on a hybrid mesh approach[9] introduced in the mid 1990's. Quadrilateral cells from overlapping surface grids were retracted (or blanked) until no overlap exists between any pair of quadrilateral cells. Zipper grids with a single layer of triangular cells were then generated in the gaps between the quadrilateral cells. Integration of surface loads was performed on the hybrid mesh consisting of the collection of non-overlapping quadrilateral and triangular cells. This algorithm was implemented into a software suite called FOMOCO Utilities[10] which contained the MIXSUR module for hybrid mesh generation and the OVERINT module for loads integration.

As the complexity of the geometric configurations increased since the introduction of MIXSUR, two major weaknesses of the software emerged. First, the triangulation algorithm lacked robustness in regions where there were large discrepancies of grid spacings in the overlapped zones. This forced the user to iteratively modify input parameters until a satisfactory hybrid mesh was produced. Second, due to an inefficient search algorithm for interpolation stencils, and the repetitive need to recompute such stencils, the code execution time became significant for cases with a large number of points in one or more surface grids.

An effort to circumvent the robustness issue of the hybrid mesh method was initiated in the early 2000's, resulting in a new method based on panel weights.[11,12] In the panel weights approach, integration of surface loads is performed on all quadrilateral cells (including the overlapped cells) where each quadrilateral cell is assigned a panel weight between 0 and 1. For any pair of overlapping quadrilateral cells, a local polygon clipping algorithm is used to determine the area of the overlap region. One of the two cells receives a panel weight of 1, while the other cell's panel weight is given by the non-overlap area divided by the cell area. The algorithm was first implemented in the POLYMIXSUR software,[11] and later independently implemented in the USURP[12] software. While the weighted panels method has been successfully utilized in a number of

*Computer Scientist, AIAA Senior Member

recent applications, the integration error introduced by the method for node-centered flow solutions has not been carefully analyzed.

The work presented in this paper begins with a local one-dimensional error analysis to compare the hybrid mesh and weighted panels methods for node-centered flow solutions. This is followed by a global one-dimensional error analysis to quantify the typical size of these errors. New developments in algorithm and software that significantly improve the speed and robustness of the hybrid mesh approach are then described. These include a faster stencil search algorithm, a more robust triangular cells generation scheme, and a more robust local surface normal determination for each triangular cell. The new algorithms are all implemented into a new version of MIXSUR (version 2.0) which has been completely rewritten in Fortran 90. Results obtained by applying the new software on complex configurations are presented in the Results Section of this paper.

II. Local Error Analysis for Hybrid Mesh and Weighted Panels Methods

In order to provide a quantitative comparison between the hybrid mesh and the weighted panels approaches, it is instructive to consider a one-dimensional local error analysis. Given a general nonlinear function $y = f(x)$, its integral over a prescribed domain in x is to be computed. For surface loads computation, this function may represent the pressure differential $-(p - p_\infty)$ or components of the viscous stresses. Also, consider a discretization of the domain in x by two overlapping grids. In the region of overlap, let the local cell spacing be Δx, the starting location of the first cell be α, and the overlap between the two cells be ϵ where $0 < \epsilon < \Delta x$ (see Fig. 1). Then the two cells occupy the regions $\alpha \le x \le \alpha + \Delta x$, and $\alpha + \Delta x - \epsilon \le x \le \alpha + 2\Delta x - \epsilon$, respectively. It is appropriate to assume equal cell sizes in the overlap region since this follows the best practices recommendations.[13] Moreover, for a local error analysis, it is sufficient to assume that the function varies linearly ($f(x) = ax + b$) in the vicinity of the overlapping cells.

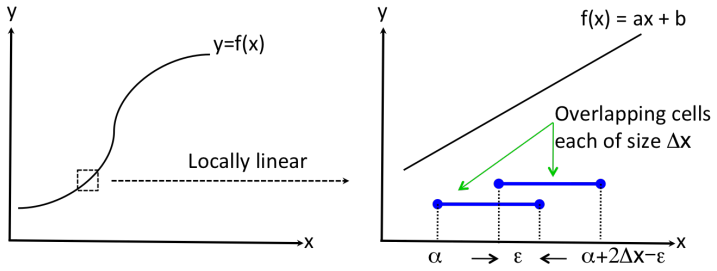


Figure 1. One-dimensional locally linear approximation to nonlinear function to be integrated.

For the local analysis, the exact analytic integral of $f(x)$ over the interval occupied by the two overlapping cells is compared to the numerical integral obtained by the hybrid mesh and weighted panels methods. From Fig. 2a, it is easily observed that the exact analytic integral I_{ex} is given by the area under the straight line from $x = \alpha$ to $x = \alpha + 2\Delta x - \epsilon$:

$$I_{ex} = \int_{\alpha}^{\alpha+2\Delta x-\epsilon} f(x)dx = (2\Delta x - \epsilon)\tfrac{1}{2}[f(\alpha) + f(\alpha + 2\Delta x - \epsilon)], \quad (1)$$

$$= (2\Delta x - \epsilon)[a\alpha + \tfrac{a}{2}(2\Delta x - \epsilon)) + b]. \quad (2)$$

For both the hybrid mesh and weighted panels methods, the numerical integral over a cell I_c is computed using the mid-point rule with

$$I_c = \frac{1}{2}(f(x_1) + f(x_2))\Delta x, \quad (3)$$

where the cell occupies $x_1 \le x \le x_2$, and $\Delta x = x_2 - x_1$ is the width of the cell, and the function values are known at the node points. Note that for a linear function f, the mid-point rule returns the area of the trapezoid in the shaded region in Fig. 2a, i.e., the mid-point rule returns the exact integral of f over the interval.

For the hybrid mesh approach integral, let the left cell be retained completely (analogous to a retained quadrilateral in the hybrid mesh), and the right cell be trimmed (analogous to a triangle in the hybrid mesh).

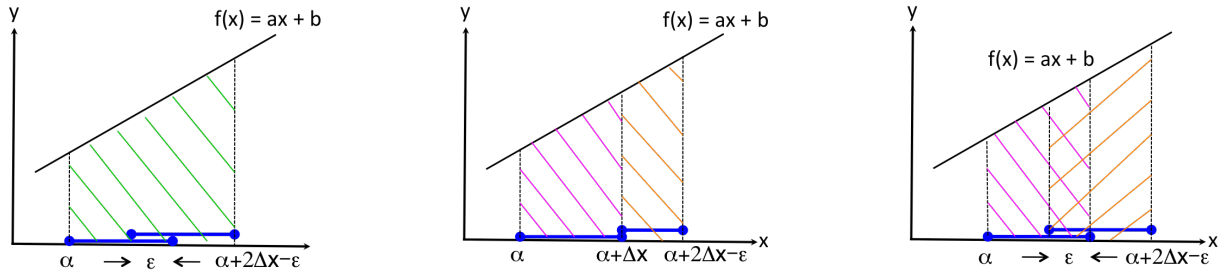


Figure 2. (a) Exact analytic integral given by area under straight line from $x = \alpha$ to $x = \alpha + 2\Delta x - \epsilon$. (b) Hybrid method integral given by sum of areas over left complete cell and right trimmed cell. (c) Weighted panels method integral given by weighted sum of areas over left and right complete cells.

It can be easily seen from Fig. 2b that the sum of of the numerical integral over the complete left cell and the trimmed right cell (or vice versa), is equal to the sum of the areas of the respective trapezoids. With the mid-point rule returning exactly these areas, the hybrid mesh numerical integral is exactly equal to the analytic integral, i.e., the local error for the hybrid mesh approach is exactly zero.

For the weighted panels approach, the numerical integral I_{wp} is given by

$$I_{wp} = I_L W_L + I_R W_R, \tag{4}$$

where I_L and I_R are the numerical integrals over the left and right cells respectively, and W_L and W_R are the weights for the left and right cells respectively. Assuming a full weight for the left cell ($W_L = 1$), and a partial weight for the right cell given by the partial cell area ($W_R = (\Delta x - \epsilon)/\Delta x$), and using the mid-point rule for I_L and I_R, we get

$$I_{wp} = \frac{1}{2}[f(\alpha) + f(\alpha + \Delta x)]\Delta x + \frac{1}{2}[f(\alpha + \Delta x - \epsilon) + f(\alpha + 2\Delta x - \epsilon)](\Delta x - \epsilon). \tag{5}$$

The absolute value of the local error for the weighted panels method E_{wp} is given by $|E_{wp}| = |I_{ex} - I_{wp}|$. After simplification, it can be shown that

$$|E_{wp}| = \frac{1}{2}|a|\epsilon(\Delta x - \epsilon). \tag{6}$$

Now, let $\theta = \epsilon/\Delta x$ be the normalized overlap parameter with $0 < \theta < 1$. Substituting into Eqn. 6 gives

$$|E_{wp}| = \frac{1}{2}|a|\theta(1 - \theta)(\Delta x)^2. \tag{7}$$

This result indicates that the magnitude of the local error for the weighted panels method is always non-zero for a non-constant function. Moreover, the local error is proportional to the local slope of the function, and to the square of the local cell size. This local error does go to zero as the cell size goes to zero, but for a finite cell size, the local error can become large for functions with high local gradients, e.g., in the vicinity of shocks, plumes, stagnation points, and shear layers.

Although the local error for the weighted panels method is non-zero, its effect on the global error, and hence on the computed forces and moments for the entire vehicle or its parts, is expected to be typically small. This is because the error occurs only in the overlapped zone which usually occupies a relatively small fraction of the total vehicle's surface area (about 1% - 3%). However, if the overlapped regions cover a larger fraction of the total surface area (as may occur in sub-components of the total vehicle), and if the flow gradients are steep in such regions (such as in the vicinity of shocks), the weighted panels method could result in non-trivial errors. Therefore, careful inspection of the results should be performed if the weighted panels method is used to compare cases with small differences in integrated forces and moments.

III. Global Error Estimate for Hybrid Mesh and Weighted Panels Methods

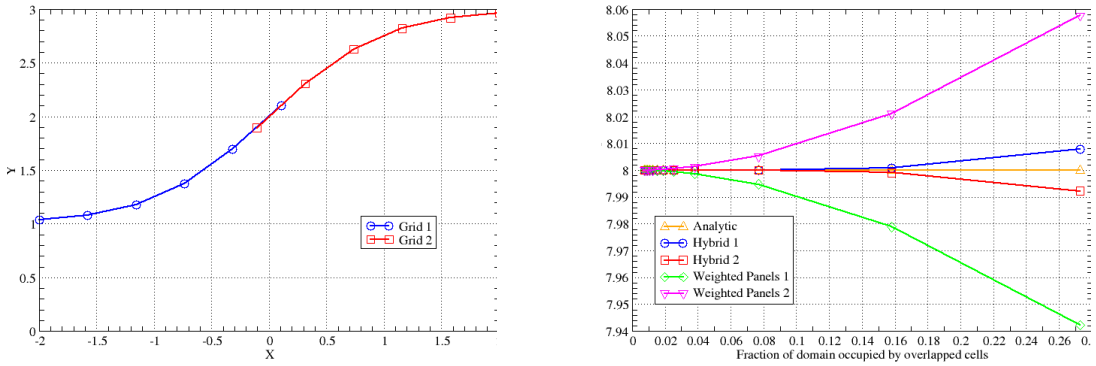
In order to assess the global error that may occur when the hybrid or weighted panels method is used to compute the forces and moments acting on a component of the vehicle, a one-dimensional global error

analysis is performed. For this analysis, a known family of analytic functions on two overlapping surface meshes is integrated analytically, and numerically using the hybrid mesh and weighted panels methods. The hyperbolic tangent function given by Eqn. 8 is chosen to represent high gradient physical variables such as the pressure near shocks, where β is a parameter that controls the steepness of the shock.

$$f(x) = 2 + \tanh(\beta x) \tag{8}$$

The domain of analysis is chosen to be from $x = -2$ to $x = 2$. Two overlapping meshes with uniform spacing are used to cover this domain where mesh 1 occupies the region $-2 \leq x \leq (\Delta x)/4$, and mesh 2 occupies the region $-(\Delta x)/4 \leq x \leq 2$, and Δx is the uniform grid spacing in both meshes. For purpose of loads integration, it is appropriate to assume that the normal double fringe overlap has been retracted, and that the two meshes overlap only between the last cell of mesh 1 (cell 1) and the first cell of mesh 2 (cell 2). In this study, it is assumed that the overlap is half a cell width ($(\Delta x)/2$). The two overlapping cells thus occupy a total width of $1.5\Delta x$. The fraction of the total domain F_o occupied by the overlapping cells is then given by $1.5\Delta x/4 = 3\Delta x/8$.

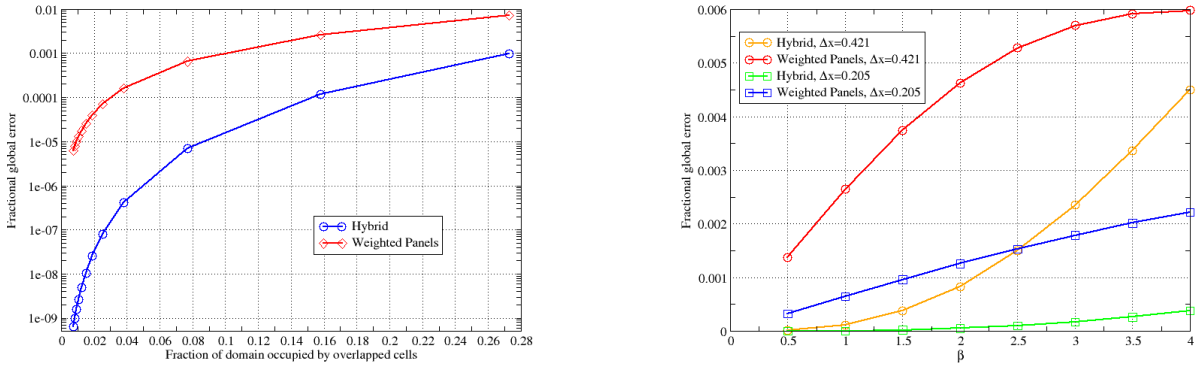
Fig. 3a shows the hyperbolic tangent function with $\beta = 1$ and 6 grid points ($\Delta x = 0.42$) on each mesh. In the non-overlapping cell region of both meshes, the numerical integral is obtained by the mid-point rule. In the overlapping cell region, the numerical integral is obtained with both the hybrid mesh and the weighted panels methods. The total integrals computed over the entire domain ($-2 \leq x \leq 2$) using the two methods are compared with the analytic integral plotted against F_o in Fig. 3b. Two results are obtained for each method. For the hybrid mesh method, result 1 is computed with the mid-point rule applied over the entire cell 1 and half of cell 2, and vice versa for result 2. Since the hybrid mesh method integrates on non-overlapping cells, the absolute quadrature error from results 1 and 2 are essentially the same as the quadrature error from a single mesh with non-overlapping cells. For the weighted panels method, result 1 is computed using a panel weight of 1 on cell 1 and 0.5 on cell 2, and vice versa for result 2. The figure indicates that the error for the weighted panels method grows quickly with increases in the domain fraction occupied by the overlapping cells (i.e., increase in Δx). Since results 1 and 2 for both methods appear to give the same absolute error, only result 1 will be used for the plots shown in Fig. 4.



(a) (b)

Figure 3. (a) Hyperbolic tangent function with $\beta = 1$, and 6 grid points on each grid ($\Delta x = 0.42$). (b) Value of integral in range $-2 \leq x \leq 2$ for analytic, hybrid mesh, and weighted panels integration, plotted against fraction of domain occupied by overlapped cells F_o ($F_o = 3\Delta x/8$).

Let I_a and I_n be the value of the analytic integral, and the value of the numerical integral, respectively. The fractional global error is defined to be $|I_a - I_n|/I_a$. Fig. 4a shows the variation in fractional global error of the total integral with respect to the fraction of the domain occupied by the overlapping cells (i.e., variation in Δx). The fractional error for the hybrid method is about one to three orders of magnitude less than that of the weighted panels method. When the overlapping cells occupy about one quarter of the domain, the weighted panels fractional error approaches 1%. Fig. 4b shows the variation in fractional error with respect to β (the steepness of the function). Again, we see that the fractional error for the hybrid method is significantly lower than that of the weighted panels method. The fractional global error from the weighted panels method remains below 1%. While some applications can easily disregard errors of this small magnitude, other applications such as airplane drag count analyses do demand more accurate accounting.



(a) (b)

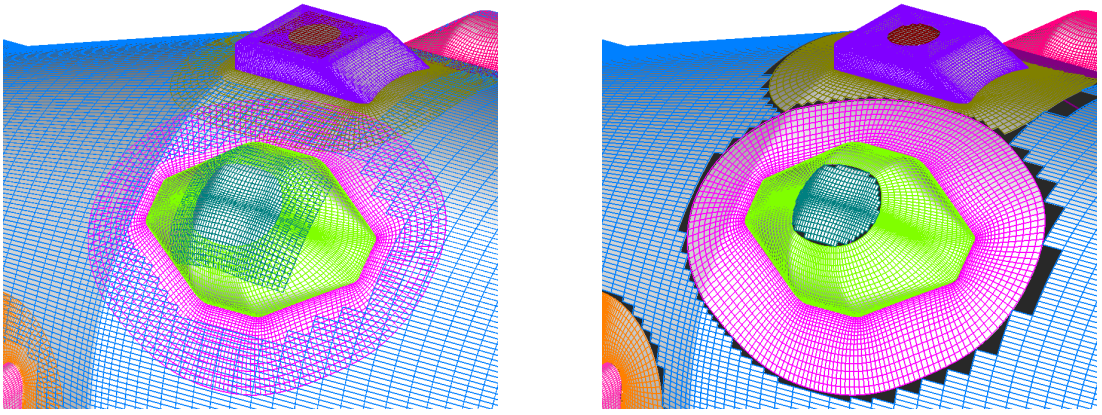
Figure 4. (a) Fractional global error versus fraction of domain occupied by overlapping cells ($\beta = 1$). (b) Fractional global error versus β.

IV. New Developments for Hybrid Mesh Method

Since the local error for the hybrid mesh approach is zero, and its global error is significantly lower than that of the weighted panels method, it could be a very attractive method if the deficiencies of its early design are removed. A completely rewritten version of MIXSUR (version 2.0) utilizing the hybrid mesh approach has been developed to address these deficiencies. Use of Fortran 90/95 allows the introduction of advanced data structures and dynamically allocated arrays that are more suitable for the algorithms needed to tackle the complex search and connectivity problems. The following subsections will describe the various enhancements made to improve execution speed and robustness.

IV.A. Quadrilateral Cell Overlap Removal Using New Stencil Search Scheme

The surface domain of the geometry is assumed to be covered by a collection of overlapping structured surface grids. The first step in the hybrid mesh method is to remove the overlap between the quadrilateral cells of such grids. Fig. 5a shows the surface grids after the domain connectivity step of grid generation. An integer iblank value for each vertex of each surface grid is inherited from the domain connectivity step. An iblank value of 1 represents a valid physical point while an iblank value of 0 represents a point that has been removed from the computational domain by the Chimera hole-cutting process. Fig. 5a shows only the quadrilateral cells where the iblank value is 1 for all four vertices.



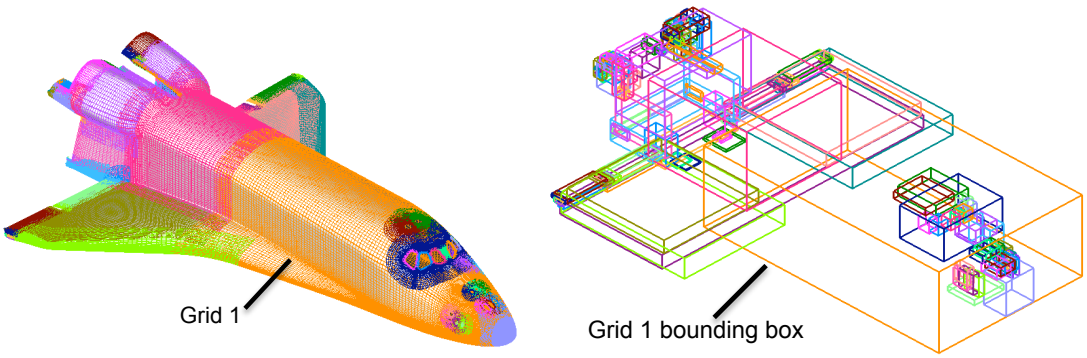
(a) (b)

Figure 5. (a) Surface grids with domain connectivity iblanks. (b) Surface grids after cell overlap removal step.

For every vertex in each surface grid, an interpolation stencil is sought from a quadrilateral cell in a neighboring surface grid. Only vertices that lie in the overlapped zones will be able to find a valid stencil. The number of vertices from each surface grid that are able to find an interpolation stencil are tallied. Between any pair of overlapping surface grids, the grid with fewer vertices in the mutually overlapping zone is labeled the coarser grid. Vertices from the coarser grid that are in the overlapped zone are then assigned an iblank value of 0. An additional step is then taken to blank out vertices in the finer grid that might still lie inside an unblanked cell in the coarse grid. This results in a retraction of the unblanked quadrilateral cells so that no overlap exists between the pairs of surface grids in question (Fig. 5b).

The above procedure requires an interpolation stencil search for all vertices of all surface grids. The previous scheme in Ref. 9 utilized an inefficient closest point search procedure where the CPU time is proportional to the square of the number of points on each surface grid. This stencil search time was typically occupying about 90% of the total CPU time for creating the hybrid mesh. For large applications such as those found in recent Space Shuttle[4] and Crew Launch Vehicle cases, the hybrid mesh generation time could take over one hour of CPU time on a single processor of a Linux Opteron (1 GHz, 1024 KB cache) desktop workstation. This was clearly unacceptable when parameter iterations were also needed in multiple runs to create a satisfactory hybrid mesh. The new search algorithm described below has been designed to produce a more rapid stencil search procedure.

The first step in the new scheme involves putting a bounding box around each surface grid. Given a point $\vec{r}_p$ seeking an interpolation stencil, the surface grid bounding boxes are used to quickly eliminate grids that are far away from $\vec{r}_p$ (Fig. 6). Once the point is determined to be inside a particular grid bounding box, there are still potentially a large number of possible donor stencils for large grids. Methods to further reduce the search time involve finer sub-division of the search space by using tree-based methods (e.g., octree, ADT-tree[14]), or other sub-bounding box methods like the structured auxiliary mesh scheme.[15]

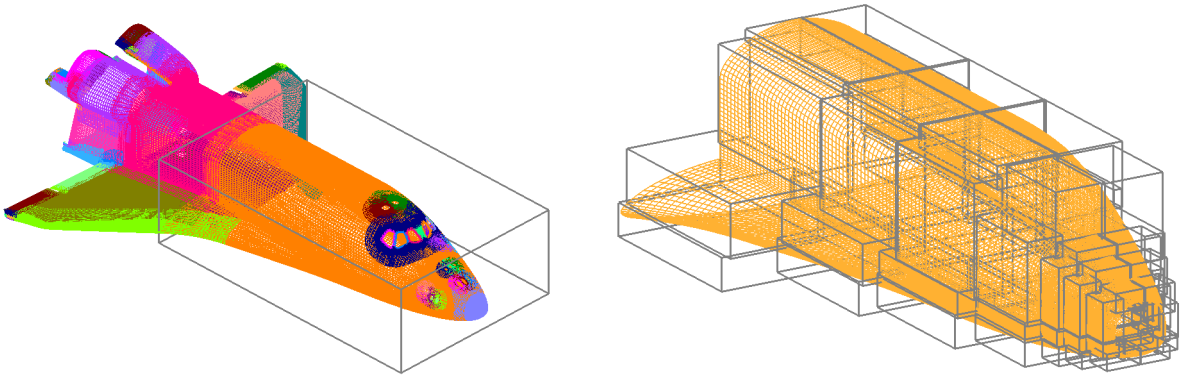


(a) (b)

Figure 6. (a) Surface grids with domain connectivity iblanks. (b) Bounding boxes of the surface grids.

In this paper, a slightly different sub-bounding box scheme is explored and is found to be very effective for the searches needed in this work. Suppose it is determined that the target point $\vec{r}_p$ lies inside a particular surface grid bounding box, where the surface grid dimensions are given by N_j and N_k in the J and K index directions, respectively. An approximately balanced sub-division of the index space $N_j \times N_k$ within this grid should provide an efficient search procedure for the interpolation stencil. This is accomplished by splitting the index space in both the J and K directions into sub-patches where each patch contains no more than N_{sp} points in both the J and K directions. Sub-patch bounding boxes are then constructed in physical space. For example, Fig. 7 shows the sub-patch bounding boxes that are created for grid 1 in Fig. 6. The next level of search for the interpolation stencil for $\vec{r}_p$ is then performed using the sub-patch bounding boxes. Once a sub-patch bounding box is identified, a closest point search within the sub-patch is used to start a stencil walk procedure to home in on the interpolation stencil. The final test involves a Newton iteration that checks whether the point $\vec{r}_p$ lies inside a particular bilinear quadrilateral cell.

Although the target point may be contained in more than one sub-patch bounding box, it is found that the sub-patch bounding boxes help to eliminate the much more expensive nearest-point search over a significant portion of the surface grid. An optimal value for the parameter N_{sp} was found to be about 25



(a) (b)

Figure 7. (a) Grid bounding box over grid 1. (b) Sub-patch bounding boxes over grid 1.

based on several numerical experiments. This value was established by decreasing N_{sp} from a large value until no effective speed-up was observed in the stencil search procedure. Decreasing N_{sp} below the optimum value results in more overhead, thus actually increasing the total CPU time. A typical speed-up for one particular case is shown in Fig. 8. Use of the sub-patch bounding box search for this case resulted in a factor of 4 reduction in total CPU time (by comparing the CPU time for large N_{sp}, essentially deactivating the sub-patch box search, and the CPU time for the optimal N_{sp}).

A further reduction in CPU time by about another factor of 10 was achieved by storing the interpolation stencils of all surface grid points in the overlapping zones instead of recomputing them when needed. The combination of utilizing both the sub-patch box search, the interpolation stencil storage, and various other optimization techniques resulted in a complete hybrid mesh creation speed-up by about a factor of 40 in the new MIXSUR 2.0 software.

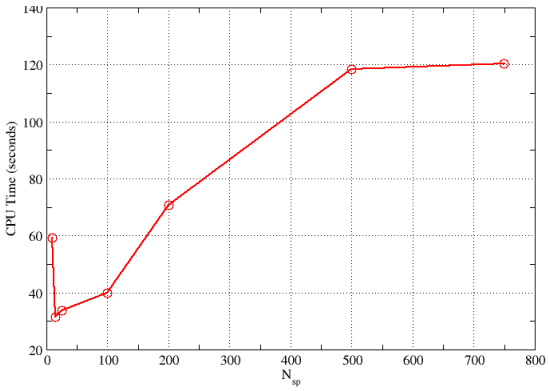
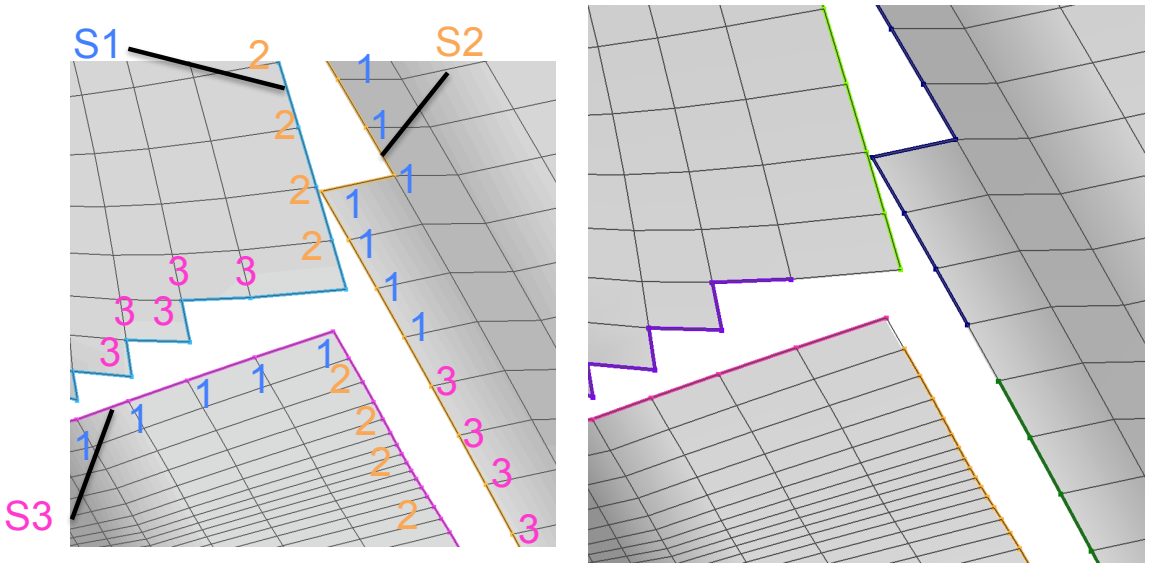


Figure 8. Variation of total CPU time for hybrid mesh approach versus number of points in one direction of sub-patch N_{sp} used to build sub-patch bounding boxes.

IV.B. Gap Boundary String Identification and Splitting

After the removal of overlap between quadrilateral cells (using iblank tagging), the next step is to identify the grid points on the boundary of the resulting gaps between the quadrilateral cells. Since only just enough vertices are blanked to remove overlap between adjacent quadrilateral cells, the resulting gaps are no larger than the size of one cell. A scheme similar to that used in Ref. 9 is utilized to automatically locate the gap boundary points by picking outer boundary points of each surface grid that found an interpolation stencil, and interior points of each surface grid that are adjacent to a blanked cell. These gap boundary points are then automatically connected to form a set of gap boundary strings by connecting along edges of the quadrilateral cells (Fig. 9a).



(a) (b)

Figure 9. (a) Three gap boundary strings, S1(blue) , S2 (orange), and S3 (magenta), are shown in colored lines before sub-string splitting. The nearest neighboring string ID number for each vertex on S1, S2, and S3 are shown as single numbers 1, 2, and 3. (b) Result of sub-string splitting based on neighboring string ID where each sub-string is identified by a different color.

Each gap boundary string can be identified by an integer ID tag. For each vertex on each gap boundary string, the nearest vertex on a neighboring string is located. Both the neighboring string ID tag and the nearest vertex index are then stored (Fig. 9a).

The current work improves the robustness of the gap triangulation process by splitting each gap boundary string into sub-strings according to two criteria. First, a gap boundary string is split at a vertex where the neighboring string ID changes from one to another. For example, string S2 in Fig. 9a is split at the vertex where the nearest neighboring string ID changes from 1 to 3, resulting in the sub-strings shown in Fig. 9b. Similar splitting is also applied to strings S1 and S3. Second, a periodic gap boundary string is split at vertices that face the start/end point of any neighboring string (Fig. 10). This results in non-periodic sub-strings that can be matched to each other more robustly (see next section).

IV.C. Triangulation of Gaps Between Quadrilateral Cells

After the gap boundary strings have been split into sub-strings, a matching procedure is performed to pair up sub-strings for triangles construction. The requirements for the triangle generation scheme are that the triangles should conform to the local geometry (especially the sharp edges), that there should be no overlap between each triangle and any neighboring quadrilateral and triangular cells, and that no new vertices are introduced in the construction of the triangular cells.

The triangle construction process is divided into three steps: self zipping, cross zipping, and polygon zipping. In the first step (self-zipping), an attempt is made to construct triangles by connecting vertices from within each sub-string (Fig. 11). A test is performed to ensure that no vertices from the matched neighboring string is contained in the constructed triangle (Fig. 11b).

In the second step (cross-zipping), pairs of sub-strings from neighboring gap boundary strings are matched. The matching criterion uses a best match of the neighboring string ID tag stored at each vertex of each sub-string. For each pair of matched sub-strings S_A and S_B, connections are made between the vertices of each sub-string to form triangular cells. The connections begin at one end of the string pair, and a front travels down to the other end as the triangles are constructed (Fig. 12).

At each location of the front, the following sequence of tests is performed to determine the appropriate next connection: either connecting point A to B^+ or point B to A^+. If connection is made to point A^+, then it will become the next point A after the front is updated, and similarly for points B^+ and B. In any step in

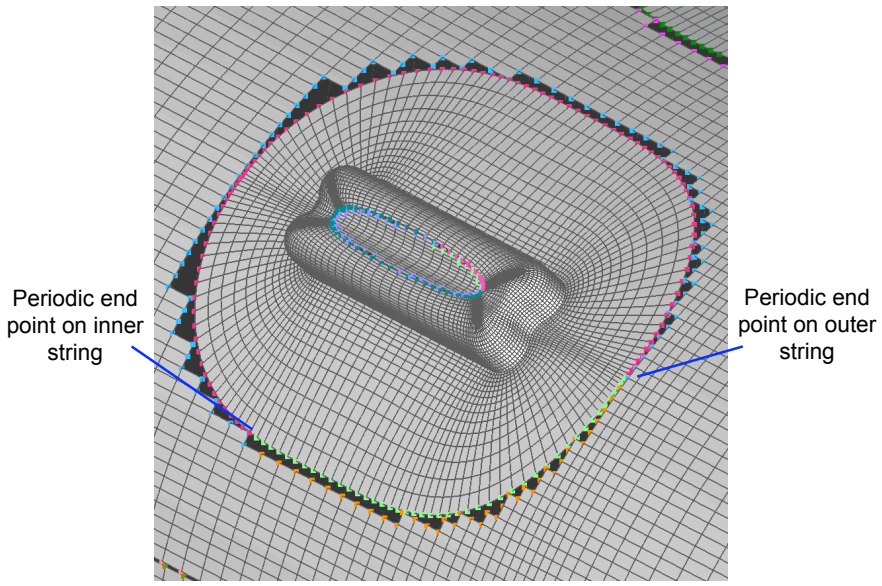
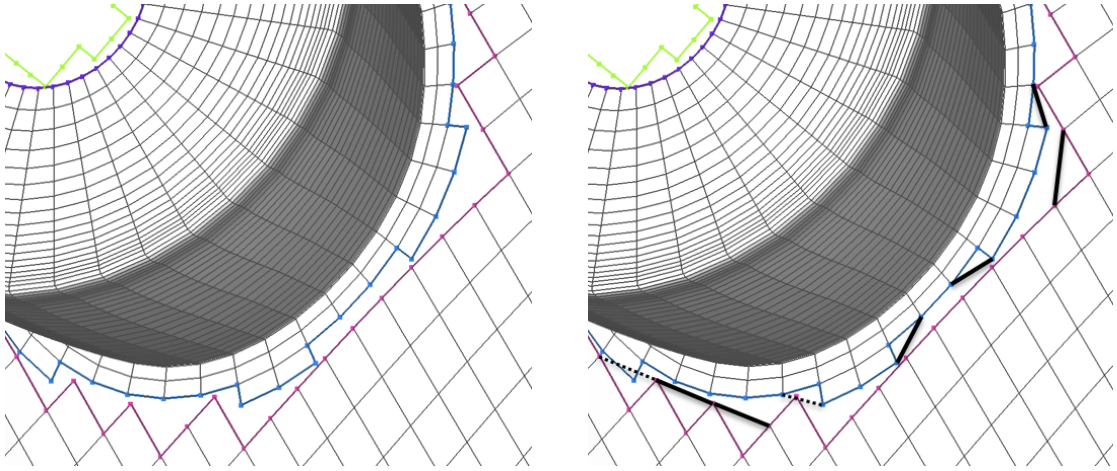
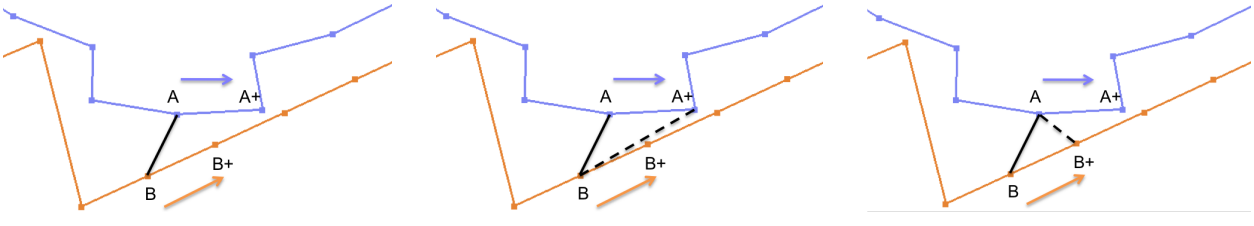


Figure 10. Result of sub-string splitting between a pair of matching periodic gap boundary strings.



(a) (b)

Figure 11. (a) Gap between unblanked quadrilateral cells before sub-string self-zipping step. (b) Solid black lines indicate triangular cells created by self-zipping step. Dotted black lines indicate triangles that are considered for self-zipping but not created since they contain a point from the matched neighboring string.



(a) (b) (c)

Figure 12. (a) Zipper grid triangle creation from matched boundary sub-strings S_A and S_B. Current front is at line segment AB. Next triangle to be created could connect to point A^+ or B^+. Arrows indicate direction of traveling front on each boundary sub-string. (b) Triangle creation by connecting to point A^+. (c) Triangle creation by connecting to point B^+.

the test sequence, if the option to connect to point A^+ or B^+ is found to be unacceptable, the test sequence is halted at the step, the front is updated, and the algorithm jumps to consider the next connection.

1. *Point-in-triangle test*

This test considers the triangle ABA^+ and determines if any neighboring points on sub-strings S_A and S_B are contained inside the triangle. If the test is positive, the connection to point A^+ is rejected. The same test is repeated for triangle ABB^+ to determine if the connection to point B^+ is acceptable (Fig. 13).

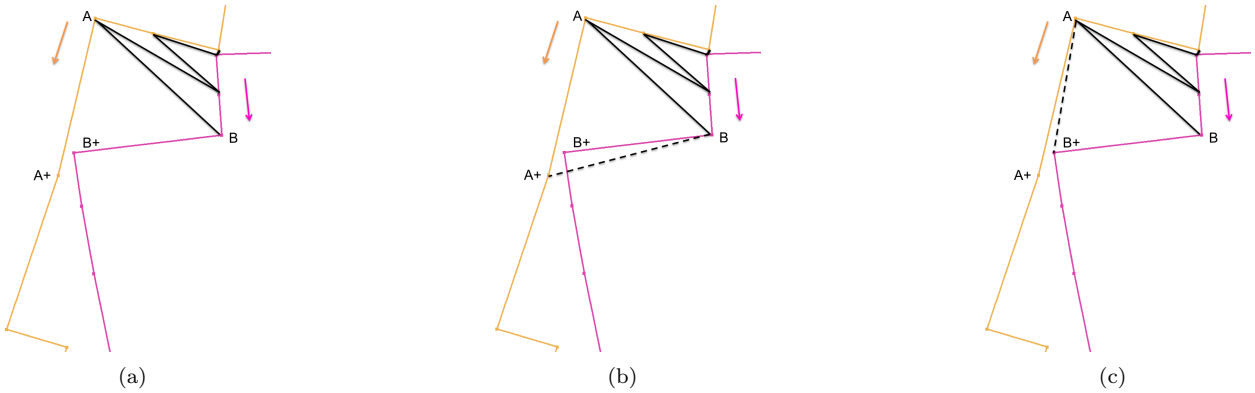


Figure 13. (a) Point-in-triangle test from front AB. (b) Connection to point A^+ is rejected since the resulting triangle contains point B^+. (c) Connection to point B^+ is allowed since the resulting triangle does not contain any neighboring vertices.

2. *Convex quadrilateral test*

This test considers the quadrilateral ABB^+A^+ and determines if it is convex. For connection to point A^+ to be valid, the vector areas of triangles ABA^+ and BB^+A^+ should be of the same sign. For connection to point B^+ to be valid, the vector areas of triangles ABB^+ and AB^+A^+ should be of the same sign (Fig. 14)

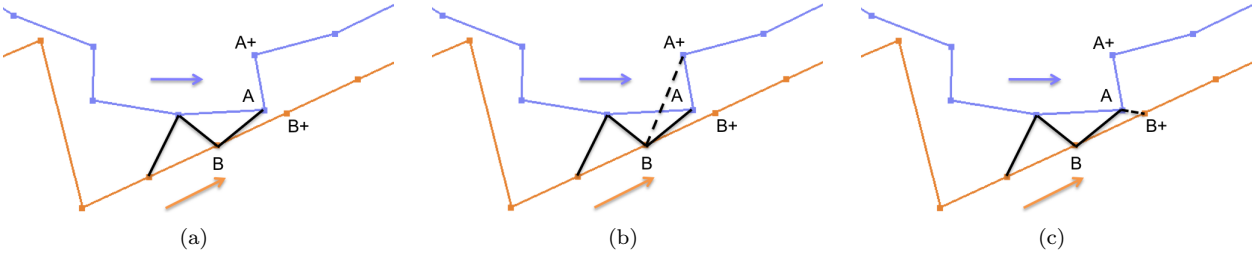


Figure 14. (a) Convex quadrilateral test from front AB. (b) Connecting to point A^+ is rejected since the vector areas of triangles ABA^+ and BB^+A^+ are of opposite signs. (c) Connection to point B^+ is valid since the vector areas of triangles ABB^+ and AB^+A^+ are of the same sign.

3. *Prism volume test*

Since the surface grids used for forces and moments integration originated from a volume grid system, it is possible to find the volume grid point that is one point away from the surface in the surface normal direction. For each triangle under consideration, the adjacent layer of points in the normal direction are used to compute the volume of the corresponding prism. If this volume is non-positive, the connection that forms the triangle is rejected.

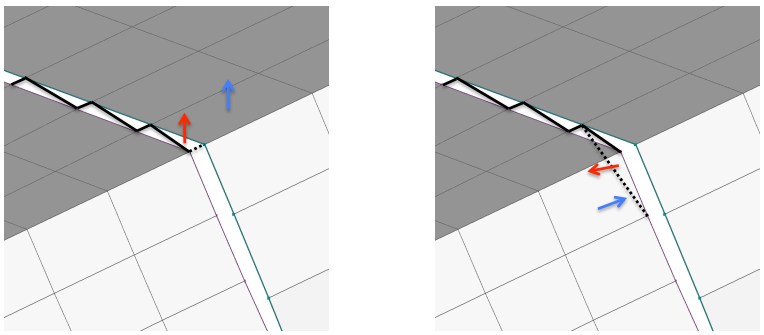
4. *Interpolation stencil test*

The local interpolation stencil test involves using the interpolation stencil for the grid points on the front to assist in the selection of the next connection. Connections between vertices that belong to the same interpolation cell (quadrilateral) are chosen over other options.

5. *Surface normal compatibility test*

For any triangle created in the cross-zipping process, there is always one edge that comes from one of the gap boundary sub-strings. This means that this same edge also bounds one of the quadrilaterals from one of the original structured overlapping surface grids in this region. A local unit surface normal can then be

computed from this quadrilateral $\vec{n}_q$. The unit surface normal for the proposed triangle $\vec{n}_t$ should then be pointing in almost the same direction as $\vec{n}_q$. In the current software, the A^+ connection choice is eliminated if the resulting triangle satisfies $\vec{n}_q \cdot \vec{n}_t < 0.99$. A similar test is also performed on the B^+ connection. This test is used to ensure that the triangles created are preserving local sharp edges of the original structured surface mesh (Fig. 15).

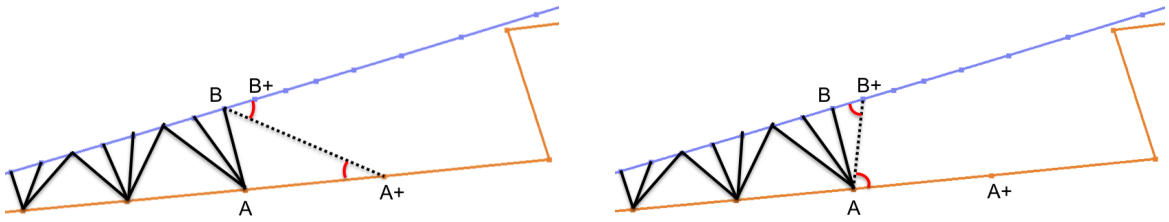


(a) (b)

Figure 15. Triangle creation choice determined by local surface normal compatibility test (dotted line indicates connection to create proposed triangle, blue arrow indicates surface normal of neighboring quadrilateral, red arrow indicates surface normal of proposed triangle). (a) Preferred choice with compatible surface normal directions between triangle and neighboring quadrilateral. (b) Rejected choice with incompatible surface normal directions between triangle and neighboring quadrilateral.

6. *Front angle test*

All tests performed so far, with the exception of the interpolation stencil test, are to prevent the creation of triangles that are topologically incorrect or are incompatible with the existing geometry and surface grids. This final test is employed to influence the connection choice to produce better quality triangles. The strategy here is to try to keep the propagating front as normal to the two boundary sub-strings S_A and S_B as possible. First, the angle between the proposed new front and segment AA^+, and the angle between the proposed new front and segment BB^+ are computed (Fig. 16). The sum of these two angles provides a measure of the orientation of the front relative to the boundary sub-strings. The connection option that results in a new front with the larger sum angle (more normal to the boundary sub-strings) is the preferred choice (Fig. 16b).

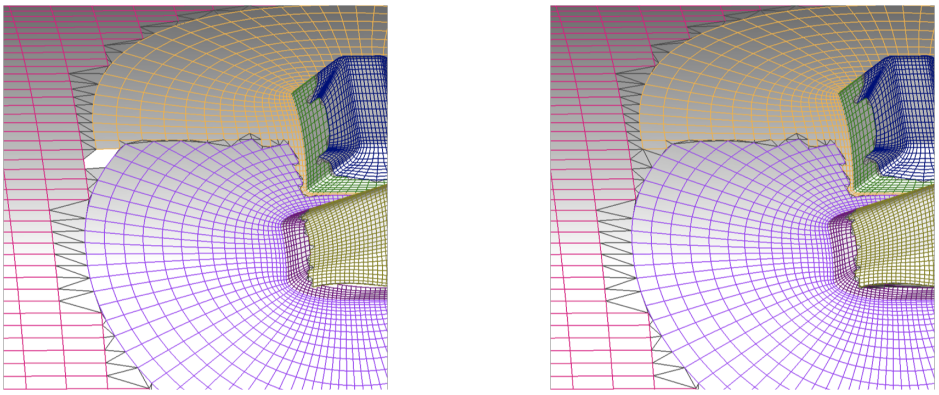


(a) (b)

Figure 16. Triangle creation choice determined by front angle test (dotted line indicates position of proposed new front, red arcs indicate angles used to determine connection choice). (a) New front with smaller angle sum to boundary sub-strings means this is not the preferred choice. (b) New front with larger angle sum to boundary sub-strings means this is the preferred choice.

After the cross-zipping step, regions where three or more pairs of gap boundary strings meet are still uncovered (Fig. 17a). The third and final step (polygon zipping) is to close up such simple polygonal regions with triangular cells. This step begins by collecting all un-used edges from all sub-strings, and then connecting these edges to form one or more closed loops (polygons). Such polygons in three-dimensions are then triangulated (Fig. 17b). Since the number of vertices on these simple polygons is typically small (less than 10), it is sufficient to use a basic recursive algorithm like ear-clipping to perform the triangulation. In the ear-clipping approach, the vertex with the smallest interior angle is first found. If the triangle formed by

this vertex and its two immediate neighbors on each side does not contain any other vertices of the polygon, this triangle is accepted. The algorithm then recursively searches for the next appropriate vertex to build the next triangle until the polygon is completely triangulated.



(a) (b)

Figure 17. (a) Triangular cells created after cross-zipping step. A polygonal gap remains in the region where three or more strips of cross-zipped triangles meet. (b) Triangular cells created after polygon zipping step.

IV.D. Determination of Triangle Vertex Ordering and Surface Normal

The vertex ordering of a triangle determines whether the surface normal is pointing out from one side or the opposite side of the triangle. In this work, the convention used is that the surface normal is given by the cross product between the vector from vertex 1 to vertex 2, and the vector from vertex 1 to vertex 3.

The direction of the surface normal of each constructed triangle in the zipper grid region has to be consistent with the surface normals of the neighboring quadrilateral cells. Since only one layer of triangles exists between the unblanked quadrilateral cells, each triangle must share at least one edge with a quadrilateral cell. Since all quadrilateral cells here are part of a structured surface grid, each cell has a defined J and K index direction that governs the direction of the quadrilateral cell surface normal. Each edge of the quadrilateral cell can then be associated with a direction so that the cross product between any two adjacent directed edges gives the same sense for the surface normal (Fig. 18). The sense of the surface normal of any neighboring triangle could then be made consistent by ensuring that the directed edge on the triangle side runs in the opposite direction to the directed edge on the quadrilateral side. The advantage of this scheme is that it only involves logical operations and not floating point operations, and hence can be made very robust.

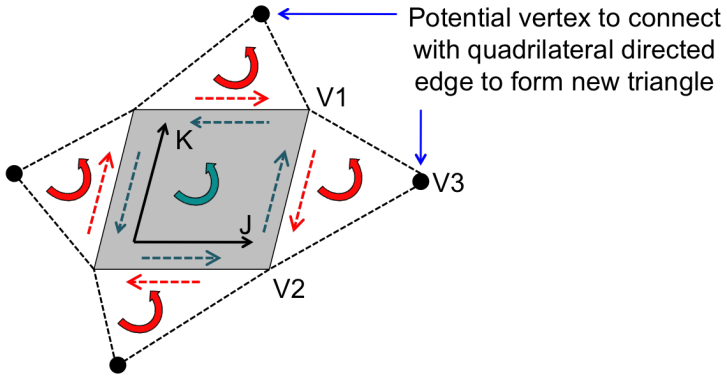


Figure 18. Determination of triangle vertex ordering (and hence its surface normal direction) using directed edge from neighboring quadrilateral cell. (V1, V2, V3) indicates vertex ordering for triangle on the right.

V. Sample Results

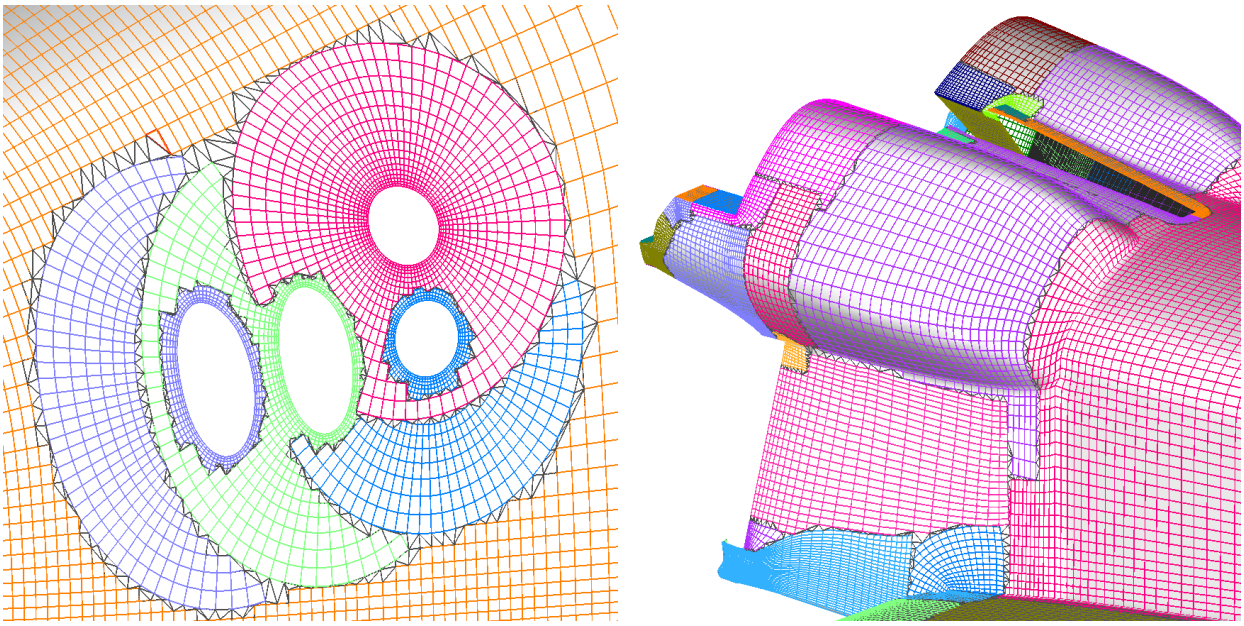
Sample hybrid meshes created using the new scheme described in this paper are presented in this section. No parameter iterations were needed to generate the final results which were computed on a single processor of a Linux Opteron workstation with 1 GHz clock speed and 1024 KB of cache.

The first case involves the hybrid mesh creation for the Space Shuttle Orbiter excluding the base region. Fig. 19a shows the region where multiple grids from the different reaction control jets overlap in the front part of the Orbiter. Fig. 19b shows the region around the OMS pod. This subset of the Orbiter geometry contains 130 overlapping surface grids and about 170,000 surface grid points. Generation of the hybrid mesh with MIXSUR 2.0 took about 4 seconds of CPU time.

The second case involves the hybrid mesh creation for the External Tank (ET) with numerous overlapping surface grids from the collars of the various attach hardware (Fig. 20). The ET system contains 109 overlapping surface grids and about 350,000 surface grid points. Generation of the hybrid mesh with MIXSUR 2.0 took about 8 seconds of CPU time.

The third test case involves the hybrid mesh creation for the entire Space Shuttle Launch Vehicle with all protuberances. The surface grid system contains 978 grids and 1.9 million surface grid points. Generation of the hybrid mesh with MIXSUR 2.0 took about 19 seconds of CPU time. A sample of the protuberances involved are shown in Fig. 21.

MIXSUR 2.0 has also been applied to various configurations of NASA's Ares-I and Ares-V rockets. The Ares-I geometry consists of a launch abort system (LAS) with nozzles, an upper-stage, an inter-stage, a first-stage, various system tunnels, feedlines, separation and tumble motors, roll-control motors, reaction control systems, cameras and antenna covers, etc. Except for the LAS, similar geometric complexities also exist for the Ares-V. For proprietary reasons, figures of the geometry and grids are not shown. The Ares-I surface grid system contains close to 200 grids and over 2 million grid points. Generation of the hybrid mesh with MIXSUR 2.0 took about 22 seconds of CPU time.

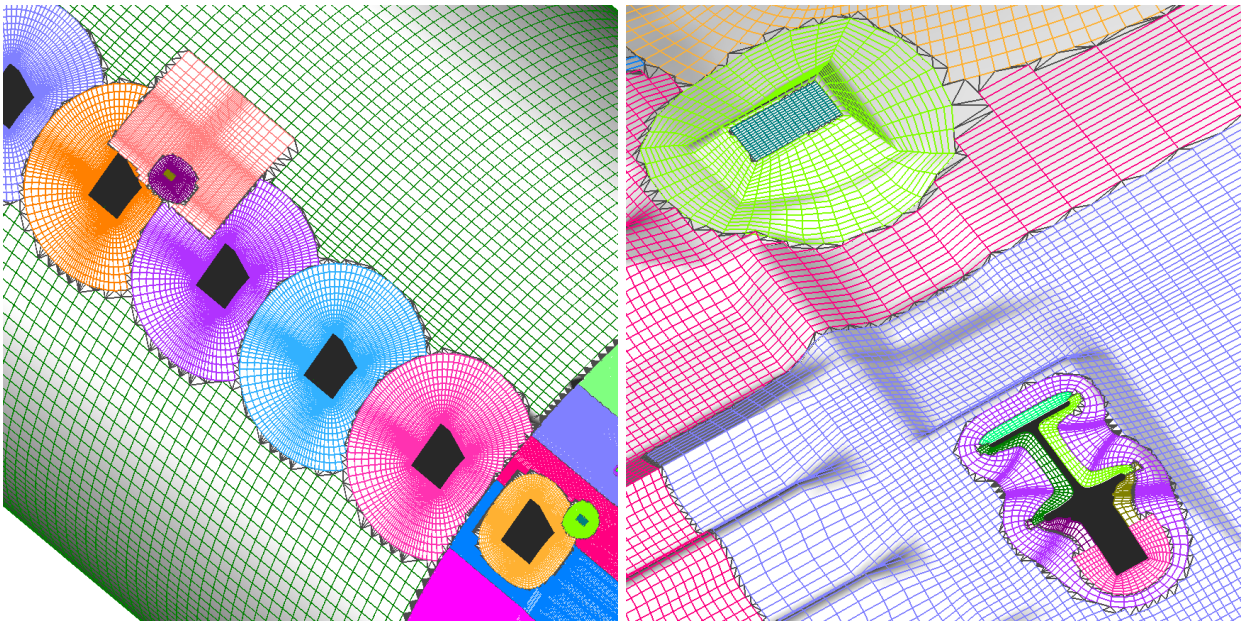


(a) (b)

Figure 19. Hybrid surface mesh created using new scheme in MIXSUR 2.0 for various Orbiter regions. (a) Reaction control jets in forward side fuselage. (b) OMS pod region.

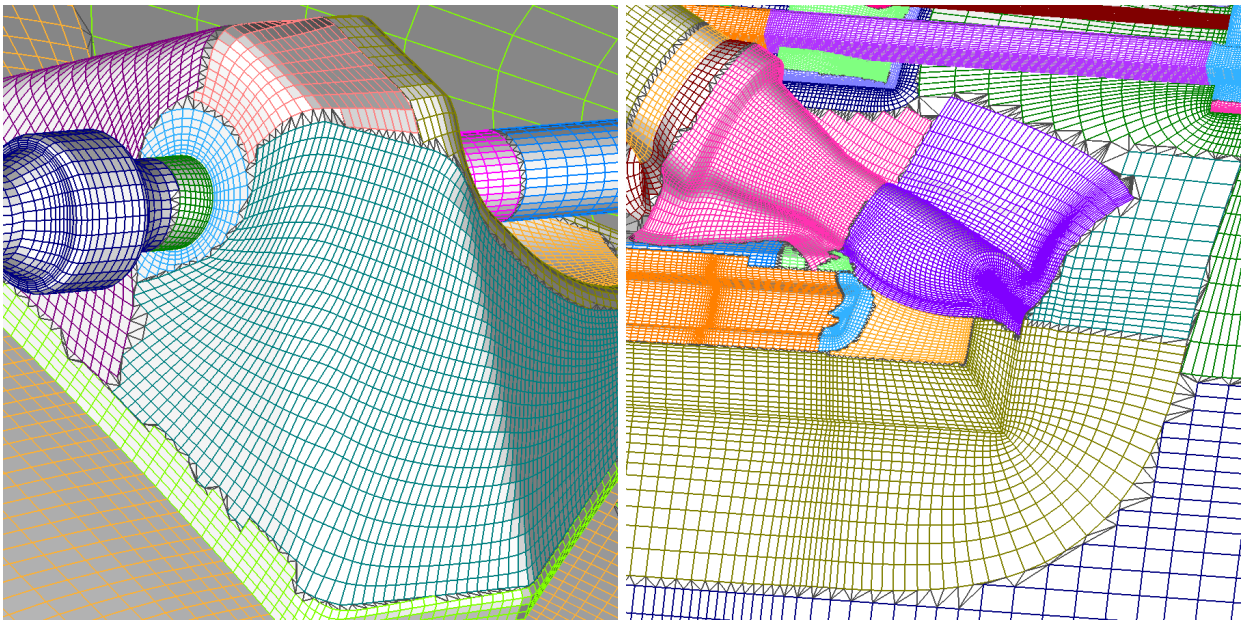
VI. Summary and Conclusions

A local 1-D linear analysis was used to assess the integration error for the hybrid mesh and weighted panels methods for performing surface loads computation on overset grids. It was found that the hybrid



(a) (b)

Figure 20. Hybrid surface mesh created using new scheme in MIXSUR 2.0 for various External Tank regions. (a) Ice frost ramps attachment region. (b) Bi-pod attachment region



(a) (b)

Figure 21. Hybrid surface mesh created using new scheme in MIXSUR 2.0 for various attach hardware for the Space Shuttle Launch Vehicle. (a) Aft-most ice-frost ramp and feedline. (b) ET thrust strut region.

mesh method has zero local error while the weighted panels method has a non-zero local error that scales with the local slope and the square of the cell size. Hence, careful analysis is advised when the weighted panels method is used in regions of high flow gradients such as shocks, plumes, and shear layers.

Results from a global 1-D analysis on a shock-imitating test function indicate that the global error of the hybrid mesh method is about one or more orders of magnitude smaller than that of the weighted panels method. However, even when the overlapped zone occupies a large fraction of the total area, the global error of the weighted panels method remains below about 1%. Whether errors of this magnitude is significant is dependent on the sensitivity requirements of the application.

Various enhancements have been developed to improve the speed and robustness of the hybrid mesh method. A new interpolation stencil search procedure plus other code-optimization techniques resulted in a speed-up of about 40 times compared to the old algorithm and software. Introduction of sub-string splitting of the gap-boundary strings, a new sequence of tests for triangles construction, and a logical test for the local surface normal of each created triangle resulted in a more robust triangular cell creation algorithm. The improved procedures have been implemented into a completely rewritten new version of the MIXSUR software. The new software has been successfully applied to a range of mission-critical complex configurations resulting in significant savings in aerodynamic loads computation time.

VII. Acknowledgements

The author would like to thank Dr. Shishir Pandya of NASA Ames Research Center, and Dr. Ralph Noack of the Applied Research Laboratory at the Penn State University for instructive discussions on the triangulation scheme. Also, the author is grateful to Darby Vicker and Ray Gomez from NASA Johnson Space Center for providing the challenging Space Shuttle test case. This work has been performed partially under NASA's Simulation Assisted Risk Assessment (SARA) group and partially under NASA's Launch and Ascent Vehicle Analysis (LAVA) group. Under these projects, the work supports aerodynamics analysis, and ground and launch operations for the next generation space exploration vehicles which include both Ares-I and V.

References

[1]Meakin, R. L., "Moving Body Overset Grid Methods for Complete Aircraft Tiltrotor Simulations," AIAA Paper 1993–3350, 1993.

[2]Slotnick, J. P., Kandula, M. and Buning, P. G., "Navier-Stokes Simulation of the Space Shuttle Launch Vehicle Flight Transonic Flowfield Using a Large Scale Chimera Grid System," AIAA Paper 1994–1860, 1994.

[3]Rogers, S. E., Roth, K., Cao, H. V., Slotnick, J. P., Whitlock, M., Nash, S. M. and Baker, M. D., "Computation of Viscous Flow for a Boeing 777 Aircraft in Landing Configuration," *J. of Aircraft*, Vol. 38, No. 6, pp. 1060-1068, Dec. 2001.

[4]Gomez, R. J., Vicker, D., Rogers, S. E., Aftosmis, M. J., Chan, W. M., Meakin, R. L. and Murman, S., "STS-107 Investigation Ascent CFD Support," AIAA Paper 2004–2226, 2004.

[5]Ray, E., "Authorizing SLAM-ER Use from the P-3C with CFD," AIAA Paper 2005–0845, Jan. 2005.

[6]Pandya, S., Onufer, J., Chan, W. and Klopfer, G., "Capsule Abort Recontact Simulation," AIAA Paper 2006–3324, 2006.

[7]Sclafani, A. J., Vassberg, J. C., Harrison, N. A., DeHaan, M. A., Rumsey, C. L., Rivers, S. M. and Morrison, J. H., "Drag Prediction for the DLR-F6 Wing/Body and DPW Wing Using CFL3D and OVERFLOW on an Overset Mesh," AIAA Paper 2007–0257, Jan. 2007.

[8]Dimanlig, A., Meadowcroft, E., Strawn, R. and Potsdam, M., "Computational Modeling of the CH-47 Helicopter in Hover," American Helicopter Society 63rd Annual Forum, May 2007.

[9]Chan, W. M. and Buning, P. G., "Zipper Grids for Force and Moment Computation on Overset Grids," AIAA Paper 1995–1681, 1995.

[10]Chan, W. M. and Buning, P. G., "User's Manual for FOMOCO Utilities – Force and Moment Computation Tools for Overset Grids," NASA TM 110408, 1996.

[11]Wigton, L., "PolyMixsur, Boeing's Replacement for Mixsur," Proceedings of the 7th Symposium on Overset Composite Grid and Solution Technology, Huntington Beach, California, 2004.

[12]Boger, D. and Dreyer, J., "Prediction of Hydrodynamic Forces and Moments for Underwater Vehicles Using Overset Grids," AIAA Paper 2006–1148, 2006.

[13]Chan, W. M., Gomez, R. J., Rogers, S. E., and Buning, P. G., "Best Practices in Overset Grid Generation," AIAA Paper 2002–3191, 2002.

[14]Bonet, J. and Peraire, J. , "An Alternating Digital Tree (ADT) Algorithm for Geometric Searching and Intersection Problems," *Int. J. Num. Meth. Eng.*, Vol. 31, pp. 1-17, 1991.

[15]Khoshniat, M., Stuhne, G. R. and Steinman, D. A., "Relative Performance of Geometric Search Algorithms for Interpolating Unstructured Mesh Data," Medical Imaging Computing and Computer-Assisted Intervention, Springer-Verlag GmbH, Eds. Ellis and Peters, 2003.